ISEL
Departamento de Engenharia Eletrónica e Telecomunicações e de Computadores

Licenciatura em Engenharia Informática e de Computadores

3º Trabalho Prático

Trabalho realizado por:

Nome: António Paulino Nº 50512

Nome: Bernardo Pereira Nº 50493

Turma: LEIC23D

Docente: João Patriarca

Arquitetura de Computadores

2022 / 2023 verão

Índice

1. Definição do mapa de endereçamento

1.

	TIPO	ORGANIZAÇÃO	CAPACIDADE
#1	ROM	8Kx16	16KBytes
#2	RAM	4Kx8	4KBytes
#3	RAM	4Kx8	4KBytes

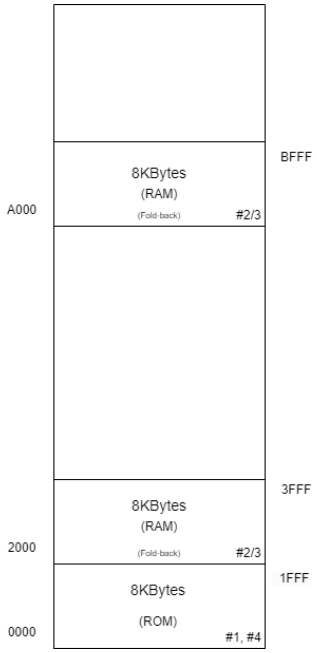
Capacidade #2 e #3 = 8KBytes

Capacidade #1 = 16KBytes

2.

	TIPO	DIMENSÃO	ACESSOS
#4	SAÍDA	2 Bytes	Word-wise

3.



Este mapa tem uma memória ROM de 8KBytes que permite leituras a words nos endereços 0000 a 1FFF e um conjunto de duas memórias ram que permite leituras e escritas a words e bytes nos endereços 2000 a 3FFF e A000 a BFFF. Para além disso existe um porto de saída acessível nos endereços 1000 a 1FFF, que está em fold-back já que é acessível através de mais que um endereço.

Existe sub-aproveitamento no dispositivo #1, já que o bit 13 do barramento de dados é utilizado na lógica do chip-select, e também é um bit de entrada nos endereços do dispositivo. Assim, o bit 13 não pode ser utilizado para mapear endereços neste dispositivo e o espaço acessível passa de 2^13 x 16 Bits = 16KBytes a 2^12 x 16 Bits = 8KBytes.
Existe Fold-Back no conjunto #2 - #3, já que o bit 15 do barramento de dados não é utilizado na lógica do chip-select, nem na entrada de endereços dos dispositivos, o que indica que este conjunto pode ser selecionado quando o bit 15 está high ou low, o que cria duas regiões.
Poderia existir conflito entre os dispositivos #1 e #4, mas como o dispositivo #4 só é utilizado em acessos de escrita e o dispositivo #1 em acessos de leitura, não existe conflito. Existiria conflito se os dois dispositivos fossem acedidos para leitura ao mesmo tempo.

4.

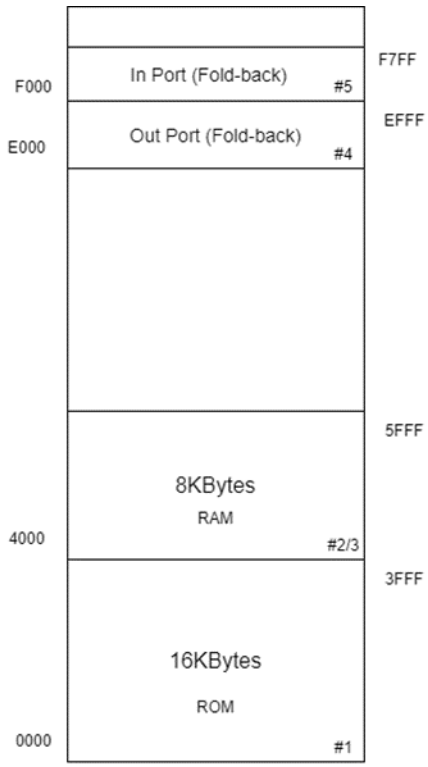
A afirmação é falsa, já que como se viu na alínea anterior, o dispositivo 1 está em sub-aproveitamento devido á partilha do bit 13 entre a lógica do chip-select, e a entrada de endereços do dispositivo, e então não é possível aceder a toda a capacidade de memória instalada no sistema.

2. Caracterização da atividade dos barramentos

	Controlo			Endereço	Dados
Instrução	nRD	nWRH	nWRL	A15…A0	D15…D0
ldr r0, sym	L	H	H	0x0000	0x0C60
	L	H	H	0x000E	0x0010
strb r2, [r1, r4]	L	H	H	0x0002	0x3A12
	H	H	L	0x1003	0x5555
push r1	L	L	L	0x0004	0x2401
push r2	L	L	L	0x0006	Z
mov r0, r15	L	H	H	0x0008	0xB780
ldr r5, [r0, #0]	L	H	H	0x000A	0x0005
	L	H	H	0x000A	0x0005
pop r3	L	H	H	0x000C	Z

3. Evolução da arquitetura

1.



Este mapa tem uma memória ROM de 16KBytes que permite leituras a words nos endereços 0000 a 3FFF e um conjunto de duas memórias ram que permite leituras e escritas a words e bytes nos endereços 4000 a 5FFF. Para além disso existe um porto de saída acessível nos endereços E000 a EFFF, que está em fold-back. e um porto de entrada acessível nos endereços F000 a F7FF, também em fold-back.

2.

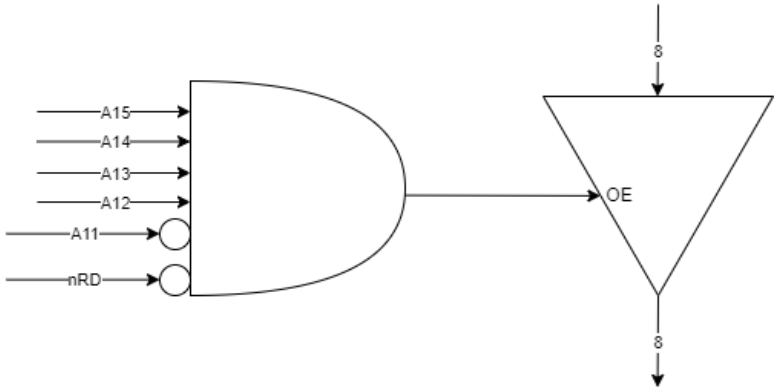
#1 CS = $\overline{A15}\ AND\ \overline{A14}$

#2,3 CS = $\overline{A15}\ AND\ A14\ AND\ \overline{A13}$

#4 EN = $A15\ AND\ A14\ AND\ A13\ AND\ \overline{A12}\ AND\ \overline{nWRL}\ AND\ \overline{nWRH}$

#5 EN = $A15\ AND\ A14\ AND\ A13\ AND\ A12\ AND\ \overline{A11}\ AND\ \overline{nRD}$

3.



Este porto de entrada utiliza a lógica de seleção da alínea anterior e um buffer tri-state de 8 bits.

4. Teste do sistema

```
.equ   ADDR_INPUT_PORT, 0xF000
.equ   ADDR_OUTPUT_PORT, 0xE000

.section .startup
       ; sec  o com c digo de arranque
       b             _start
       b             .
_start:
       ldr           sp, addr_stack_top
       mov           r0, pc
       add           lr, r0, #4
       ldr           pc, addr_main
       b             .
addr_main:
       .word main
addr_stack_top:
       .word stack_top

       .text
       ; sec  o com c digo aplicacional
main:
    bl inport_read
    lsl r1, r0, #15
    bzs main
    lsl r0, r0, #8
    asr r0, r0, #8
    bl outport_write
    b main
```

```
/* ------------------------------------------------------
 * Implementa  o de API para portos paralelos
 * ------------------------------------------------------*/
/* Devolve o valor atual do estado dos bits do porto de entrada. */
; uint8_t inport_read ();
inport_read:
       ldr    r1, inport_addr
       ldrb r0, [r1, #0]
       mov    pc, lr

inport_addr:
       .word ADDR_INPUT_PORT

/* Faz a inicialização do porto, atribuindo o valor value aos seus bits. */
; void outport_write ( uint16_t value );
outport_write:
       ldr    r1, outport_addr
       str    r0, [r1, #0]
       mov    pc, lr

outport_addr:
       .word ADDR_OUTPUT_PORT

       .data
       ; sec  o com dados globais iniciados
       ; ...

       .section .bss
       ; sec  o com dados globais n o iniciados

       .equ STACK_SIZE, 64
       .section .stack
       ; sec  o stack para armazenamento de dados tempor rios
       .space STACK_SIZE
stack_top:
```